Energy, Mines and Resources Canada | Énergie, Mines et Ressources Canada

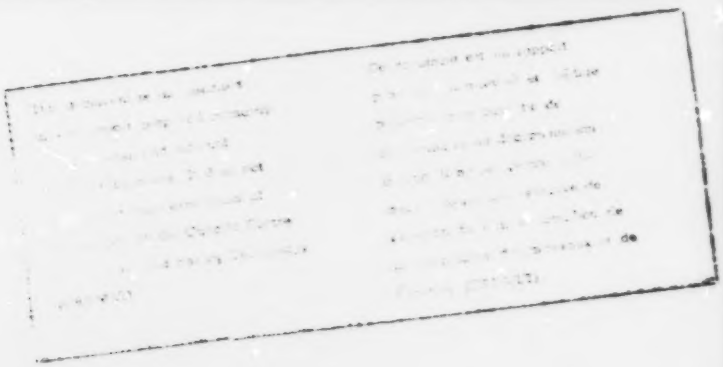
CANMET

Canada Centre for Mineral and Energy Technology | Centre canadien de la technologie des minéraux et de l'énergie

NEURAL NETWORKS AND THEIR APPLICATIONS IN MINERAL PROCESSING: PART II. WHY USE NEURAL NETWORKS AND WHERE?

J. Leung
Core Technologies and Services Laboratory

September 1992



Project: 30.50.99
Computer-Aided Mineral Processing

MICROMEDIA

MINERAL SCIENCES LABORATORIES
DIVISION REPORT MSL 92-63 (IR)

Crown Copyrights Reserved

24 pp

93-01519/

NEURAL NETWORKS AND THEIR APPLICATIONS IN MINERAL PROCESSING: PART II. WHY USE NEURAL NETWORKS AND WHERE?

by

J. Leung*

ABSTRACT

This is the second part of the report with an objective of introducing neural networks to the mineral processing industry. Having described the nature of a neural network in part I, this report will discuss the reasons behind the popularity enjoyed by neural networks and the areas where they can be applied by comparing with other technologies, such as statistical methods and expert systems. The advantages and limitations of neural networks are presented. Several examples using laboratory data and industrial plant data are given in applications such as noise filtering, modelling, simulation and data forecasting. Although this report focuses on the applications in mineral processing, the neural computing technique can be useful in other process industries as well.

Keywords: Neural networks, mineral processing, modelling, simulation, artificial intelligence

---

* Core Technologies and Services Laboratory, CANMET, Energy, Mines and Resources, 552 Booth Street, Ottawa, Ontario, K1A 0G1.

TABLE OF CONTENTS

	Page
ABSTRACT	i
INTRODUCTION	1
WHY USE NEURAL NETWORKS	1
Comparison with Statistical Models	2
Regression model	2
ARMA model	3
Comparison with Expert Systems	6
Advantages of Neural Networks	7
System identification	7
Adaptiveness	7
Fault tolerance	8
Generalization	8
Limitations of Neural Networks	8
Large quantity of data	8
Empirical parameters	9
Local minima	9
WHERE TO USE NEURAL NETWORKS	9
Signal Processing	10
Noise filtering	10
Pattern recognition	10
Data forecasting	10
Modelling and Simulation	11
Control and Optimization	13
EXAMPLES	13
Noise Filtering	13
Forecasting	18
Modelling and Simulation	21
Process background	21
Data analysis	23
Network design	26
Results	30
Pruning	34
CONCLUSIONS	36
ACKNOWLEDGEMENT	36
REFERENCES	36
APPENDIX: TIPS FOR NEURAL COMPUTING	39

TABLES

No.		Page
1.	a) Time delays determined by cross-correlation functions	29
	b) Partial auto-correlation of variables	29

FIGURES

1.	A typical regression model	2
2.	A typical network structure	4
3.	An ARMA model	5
4.	a) Forward neural model b) Inverse neural model	12
5.	Neural-based predictive control	14
6.	Feedward neural control	14
7.	A recirculation network	15
8.	Raw and filtered signals with NN and moving average	17
9.	Power spectra of raw and filtered signals	18
10.	A back-propagation network used in data forecasting	19
11.	Data forecasting where the first 200 data points are used for network training and the remaining 120 data points are predicted by network	20
12.	Modelling residuals of calculated and measured Cu concentrates	21
13.	Process flowsheet for the treatment of copper-zinc ore	22
14.	Time series of input and output variables	24
15.	Cross-correlation plots of some process variables	25

FIGURES (Cont'd)

No.		Page
16.	Cross-correlation plots of main process variables using first order difference of the time series	27
17.	Partial auto-correlation plots of main process variables	28
18.	Input/output node design for MIMO network 13-4-2	30
19.	Calculated and measured values of two output variables where the first 200 data points are used for the training of network 10-4-2 and the remaining 127 data points are simulated by the network	32
20.	Scatterplots of two output variables in simulation	33
21.	Modelling residuals of two output variables in modelling and simulation	33
22.	Modelling residuals of network 12-4-2 where the first 200 data points are used for network training and the remaining 127 data points are simulated by the network	34
23.	Modelling residuals of two output variables using network 10-4-2 with 10% pruning	35

INTRODUCTION

In recent years, neural networks have attracted a lot of attention from researchers in various fields, just as expert systems did a few years ago. While this "hype" has inspired rapid development of the technology, it also leads to two possible dangers, that of inappropriate over-expectation, or complete disbelief in the technology leading to it being ignored. With better understanding, these dangers can be avoided so that potential users will look at this technology realistically for feasible applications in their own fields. The nature of network architecture and the capability to learn have made neural networks more attractive in some applications than other techniques such as statistical methods in numerical processing or expert systems in symbolic processing. The neural computing technique is investigated, not to replace other techniques but to serve as a potential alternative tool or to incorporate with other techniques to complement each other's weakness. As with other developed technologies, neural networks have their advantages and limitations. The numerous areas where neural networks can be useful are, for example, data smoothing, noise filtering, pattern recognition, modelling, simulation and process control.

This report compares neural networks with the existing techniques of statistical methods and expert systems and introduces the concept of a hybrid system that incorporates neural networks into them. The combined technology may be a better solution for many complex problems. Various applications of neural networks are identified and a few examples using laboratory and Kidd Creek flotation circuit data are given for illustration.

WHY USE NEURAL NETWORKS

The structure of neural networks, inspired by the neuron connections of the human brain, mainly consists of many processing elements located in multi-layers of the network and interconnected to each other. The learning algorithm that is used to define linear or nonlinear relationship between the input-output data set, is based on mathematical calculations. Then why are neural networks so popular and powerful? By comparing with other technologies in the following

section, one can find the particular features and capabilities unique to neural networks and thus identify the specified areas where they are most suitable.

Comparison with Statistical Models

Regression model

Statistics is a methodology that has been well-developed and used extensively in almost every field. Although the scope covered by statistical methods is wide, only the statistical models used in defining the relationship between input and output variables will be compared here, i.e., the regression model shown in Figure 1. The model has an equation defined by:

$$Y=a_0 + a_1X_1 + a_2X_2 + ... + a_nX_n$$

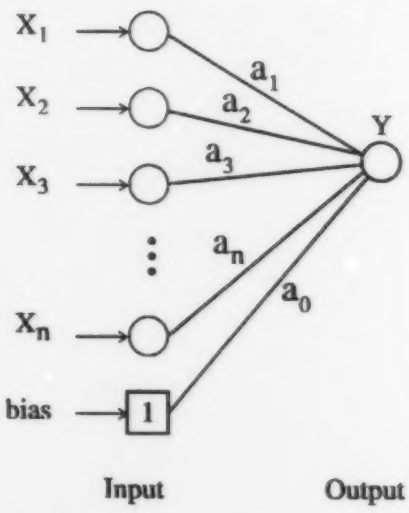


Fig. 1 A typical regression model

where the X_i are input, a_i are the coefficients related to each input and Y_i is an output. Both regression and neural network methods work for the same basic applications (1) and may be extended to include certain non-linearity and interactions between any two variables. However, when using a regression model, one should specify the exact parameter terms and even limit the number of possible interactions before any numerical computations can be performed and only one pass through the data set is required. The coefficients a_i are determined by finding the best least squares fit and their values are used to determine whether the variables X_i are significant. This explicit property makes the interpretation of the model and the performance calculations such as significant test, confidence intervals much easier. Certain assumptions on the underlying distribution of data are usually made.

The regression model may actually be represented by a two-layered linear Back-propagation neural network (2) in its input and output layers (see Figure 2). The addition of hidden layers increases the ability of better extracting the essence of the dynamic process behaviour. Besides, each processing element in hidden and output layers is passed through a nonlinear transformation, e.g., from $\mathbf{Z}_{in}$ to $\mathbf{Z}_{out}$ or from $\mathbf{Y}_{in}$ to $\mathbf{Y}_{out}$. This characteristic and the fact that there are multiple elements in the hidden layer, provide the capability of modelling more complex relationships that are nonlinear. Therefore, a Back-propagation neural network can simply be viewed as a nonlinear model and the parameters of the model are the weight connections (e.g., $\mathbf{W}_{ji}$ and $\mathbf{W}_{kj}$ in Figure 2) between elements of adjacent layers. More inputs can be examined for interactions since the network will self-select the appropriate inputs that help to predict the output through repeated learning from examples. Therefore more calculation time is required in a neural computing system than a regression model.

ARMA model

A conventional time series model, the Auto-Regressive Moving Average model (ARMA) shown in Figure 3, may also be represented by a two-layered Back-propagation network. Assuming t

is the current time and *t-1, t-2,...,t-n* are the time lags, the ARMA equation is defined as:

$$Y_t = a_0 Y_{t-1} + a_1 Y_{t-2} + \ldots + a_m Y_{t-m} + b_0 X_t + b_1 X_{t-1} + \ldots + b_n X_{t-n}$$

where the coefficients a_i, which represent the auto-regressive portion of the ARMA model, are the multipliers of past values of the output Y and the coefficients b_j, which represent the moving average portion, are the multipliers of present and past values of an external variable X. The model has exactly the same form as the regression model, except that the inputs and outputs now denote values of time-varying signals and are represented by the processing elements (PE) in the input and output layers of a neural network. The addition of hidden layers in the network and the nonlinear transfer function both increase the capability of the model.

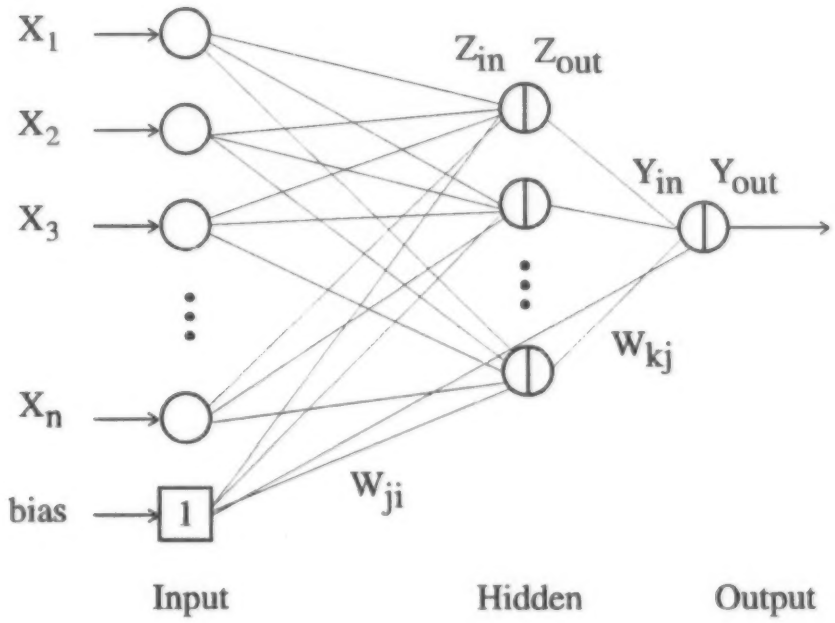


Fig. 2 A typical network structure

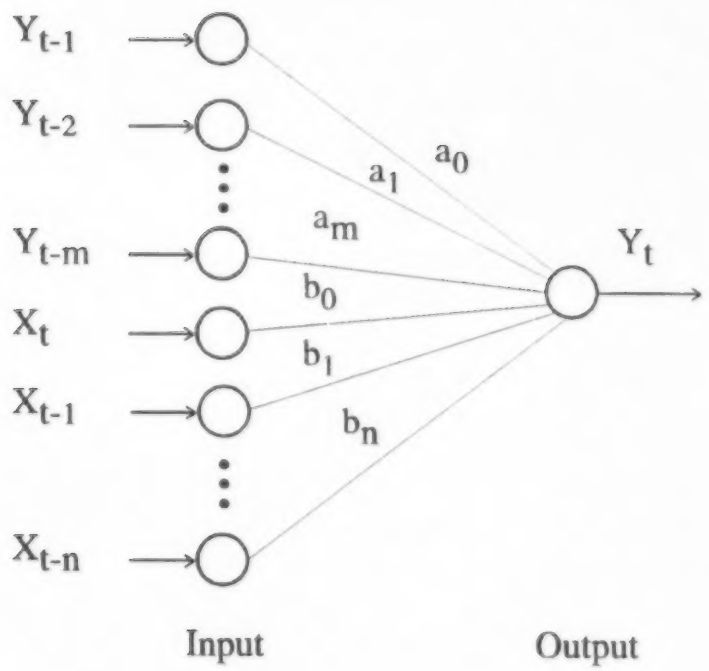


Fig. 3 An ARMA model

Instead of using neural computing techniques alone, one can incorporate statistical methodology with neural modelling, particularly in the areas of pre- and post-processing of data. For example, descriptive statistics and graphics may be used to determine explanatory variables and to understand data distributions, factorial analysis may be used to understand the complexity and interrelationships of data, or performance calculations may be used to determine the best fit of data. In other words, statistical methods may be very helpful in understanding the data, which can lead to development of a better neural network model.

Comparison with Expert Systems

Research in Artificial Intelligence (AI) started in the 1950's with the objectives of understanding and mimicking the cognitive processes of human brains. It then evolved into new disciplines such as expert systems and neural networks. Despite sharing a common starting point, expert systems and neural networks each employ a different methodology, emphasis and style. Expert systems, the black-box macroscopic intelligence, are one of the first commercial applications developed from AI which have attracted most attention in the past few years.

An expert system is defined as "an intelligent computer program that uses knowledge and inference procedures to solve problems that are difficult enough to require significant human expertise for their solution" (3). This technology may be characterized as a top-down method, where knowledge is abstracted from human experts, stored in symbolic rules in the format of phrases such as "if...then" and processed through logical reasoning paths. Therefore, an expert system must be able to capture knowledge at the level of abstraction at which humans use it, a level which includes not only facts and deductions, but also intuitions.

In contrast, neural networks may be characterized as a bottom-up method. By creating components that mimic neurons and connecting them like neurons, a system is created that can approach some of the behaviour of a biological system. Unlike expert systems whose primary requirement is a domain expert, neural networks require a data set to derive expertise. To have a network properly trained, expertise is essential in selecting representative training data and explanatory variables. An expert system is used, when relations are already known and can be clearly stated and computed. Whereas a neural network is used when the relationship is complex and not understood or when a relationship can be stated in general terms but is vague in details or difficult to compute.

In recent years, expert systems have lost part of the initial momentum mostly due to the overexpectation of the technology (4) and much attention has shifted towards neural networks. Despite the fact that the opinions of scientists tend to be split into opposing camps of neural

networks and expert systems, the two technologies are in fact complementary to each other and can be combined in various areas. The combination of the two will often be more powerful than either of them individually (5). Neural networks, either single or multiple, parallel or sequential, can be used as a front-end tool that can turn fuzzy data into categories or reliable information and based on this information, expert systems can be used to suggest actions, diagnosis or decisions. A few examples of hybrid systems with combining approaches have already been reported (6,7,8).

Advantages of Neural Networks

The advantages of using a neural network can be categorized in the following areas:

System identification

The massive interconnected structure of a neural network makes it capable of capturing the nonlinear and dynamic behaviour of a process. Unlike expert systems, a neural network can learn and self-organize itself to achieve the desired input/output mapping without actually knowing any rules. Therefore, a domain expert is not always required. The ability to select combinations of features pertinent to the problem and to deduce these features on its own also make neural network especially useful in pattern recognition applications. Furthermore, its parallel architecture makes neural networks amenable to advanced technology developments in hardware with the potential for increases in computation speed. For example, an electronic neural network (ENN) chip currently being developed by AT&T where network components are placed on a silicon chip.

Adaptiveness

Neural networks have the ability to adapt (i.e., to learn and adjust) to changes in the data and learn the characteristics of time-varying input signals without any knowledge of the exact characteristics, configurations or operating conditions of a process. This ability is very important in process control where a system will adjust its parameters automatically on-line in such a way as to compensate for changes in the dynamics of the process.

Fault tolerance

Information in a network is stored in connection weights distributed throughout the network. If a small number of processing elements are disabled, impaired or the connections are distorted, the behaviour of the network will degrade but will still function. This characteristic of fault tolerance is extremely important in the applications where breakdown of a control device often means disaster, e.g., in nuclear power plant operation or missile guidance. An example is given in this report.

Generalization

By its nature, a neural network is capable of dealing with new circumstances. When a trained network is presented with incomplete, noisy or new inputs, the network will still respond with reasonable results, as opposed to most of the other techniques. This property is called "generalization". However, the quality of generalization depends on various factors, such as the type of the application, the size and distribution of the training set available in the problem domain and the type and quality of the network model used.

Limitations of Neural Networks

Currently, neural networks have some limitations that need improvement and they are listed as follows:

Large quantity of data

One disadvantage of a neural network is the large amount of data needed in order to cover all scenarios of the operating conditions of a process. Training data that contain routine, unusual and boundary condition cases of the process should be high in quality and diverse enough to add information to the network model. Therefore, computation speed could be slow for large scale problems that require a significant amount of training data or connection weights. Besides some of the enhancements that have already been achieved to increase convergence performance and speed, more research is needed to develop advanced technologies (2) in fields such as optics to

facilitate the unconventional network computing architectures and finally to develop quantitative performance tools to evaluate the network results.

Empirical parameters

Since information is distributed throughout the network, it is very difficult to interpret the network parameters in terms of process parameters either qualitatively or quantitatively. In other words, there is no physical meaning attached to the model coefficients and this is one of the limitations for empirical modelling techniques which include regression models, transfer functions, neural networks etc. More research is needed to further investigate and understand brain information processing ability.

Local minima

Another drawback is that, due to the nonlinear functions used in the layered networks, it is difficult to tell, during the network training whether the system has reached its global minimum or is trapped in one of the local minima (2). Therefore more research is also needed to develop new algorithms that are fast, robust in the presence of noise, stable and guaranteed to converge to the global minimum.

WHERE TO USE NEURAL NETWORKS

In the processing industry, the optimal goal is to recover materials with highest quality at least cost. In mineral processing, due to the time-variation of the chemical, geometrical and mineralogical characteristics of the treated ores, the process is more versatile and thus more difficult to identify and control than most other processes. Using neural networks technology to efficiently extract useful information, to classify data, to identify process dynamics and to control parameters to achieve optimal operating conditions seems to be a natural move. The following sections will discuss a few applications and report some case studies already published in the literature.

Signal Processing

Signal processing is an important procedure to extract, forecast or classify information from raw data. The subject includes noise filtering, pattern classification and data forecasting, etc.

Noise filtering

When data are acquired from a system, they usually consist of fluctuations caused by factors other than the process itself (e.g., variability in measuring methods of a sensor) or by short-term perturbations of the process. Therefore, data collected in a process may contain noise that hide the key information. Filtering is a technique that allows one to retain the essence of the data while minimizing the effects of fluctuations. Common approaches to low pass filtering are moving average, repeated sample average and first order digital filtering. Recently, neural networks have appeared in the picture and are considered to preserve a greater depth of structure and detail of the data than traditional low pass filters, while still removing noise.

Pattern recognition

Pattern recognition tasks require the ability to match large amounts of input information quickly and then generate categorical or generalized output. They also require a reasonable response to noisy or incomplete input. Neural computing systems possess these capabilities as well as the ability to learn and build unique structures specific to a particular problem. They are thus especially useful in pattern recognition. Gorman and Sejnowski used a Back-propagation network to classify sonar targets and reported that the network performance was better than a nearest neighbour classifier and comparable to a trained human listener (9).

Data forecasting

Forecasting is an important procedure in operational planning and control especially in a process where many uncertainties could arise. Conventional forecasting requires a difficult and time consuming modelling and simulation of the process. Instead, a neural network may be trained on past values on the assumption that future values are predictable. Lapedes and Farber reported that in predicting nonlinear chaotic time series, a Back-propagation network exceeds conventional

methods by orders of magnitude in accuracy (10). Applications that used neural networks can be found in financial analysis and economical forecasting. One example reported for data forecasting is to use the closing price of the S&P 500 for ten previous weeks to forecast the closing price for the following week (11).

Modelling and Simulation

In process control, it is very important to have a sufficiently accurate model of the plant that is to be controlled. The model should be able to identify the process dynamics, ideally, in all operating conditions. In general, there are four modelling approaches: mechanistics models, phenomenological models, empirical models and non-parametric models. Each approach has its advantages and drawbacks. For example, the mechanistic models require a knowledge of physio-chemical theory in the process. Therefore the time required to develop the model can be lengthy depending on the complexity of the process. The model derived may be theoretically sound but may not be suitable in a complex industrial application. A neural network model is considered to be an empirical model which describes the quantitative and qualitive relationship that exist between observed inputs and outputs. With modern data acquisition systems, large amounts of historical data are normally saved and can be retrieved for modelling purposes.

The advantages of using a neural network model are: 1) development time is less than with most of the other methods; 2) knowledge of the process to be modelled is useful but not essential; 3) real plant data are usually available; 4) model is adaptive; and 5) good for processes that are difficult to model using first principles.

There are two types of network models, the forward neural model and the inverse neural model (12). There are differences between the two models. The forward neural model determines the output of the process **Y** from the current and past values of its input **U** and output **Y**. The inverse neural model estimates the manipulated variable **U** that will provide the desired output $\mathbf{Y}^*$ after the delay time **d** (Figure 4). One must be very careful when using an inverse model in a complex situation, e.g., in multiple inputs single output (MISO) or multiple inputs multiple

outputs (MIMO) problems. There may not be a unique solution and extensive calculations and analysis may be required as the number of degrees of freedom increases.

Similarly, neural network techniques can be used to simulate the response of a process to various input scenarios by using a trained network. One can predict the output states at the next sampling interval by presenting once only to the trained network the past and present values of the inputs and/or outputs. A neural network is generally very efficient at predicting state variables in the range of the data contained in the learning data set. However, it does not perform well when data extrapolation is required.

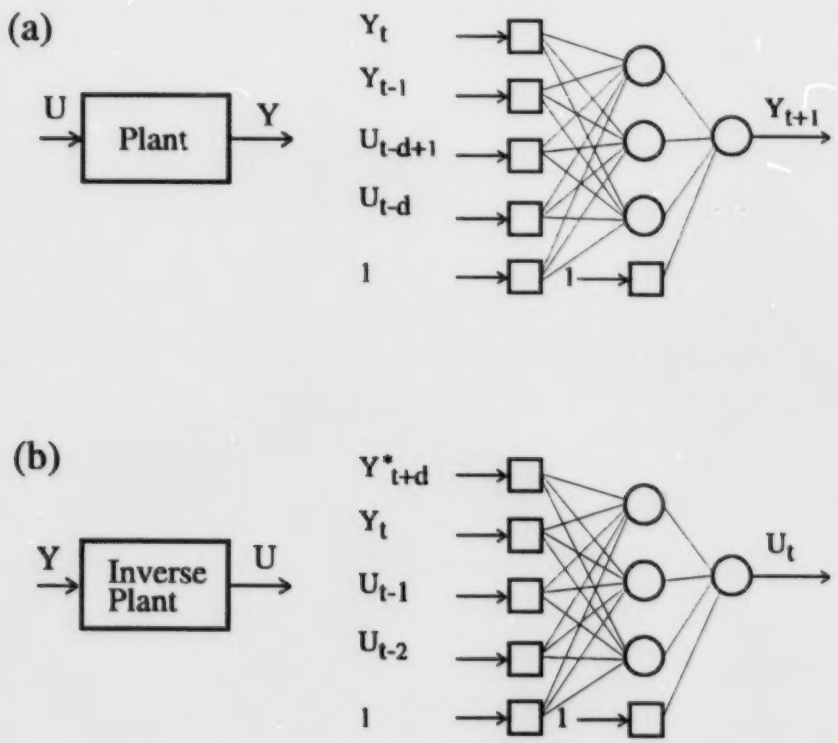


Fig. 4 a) Forward neural model b) Inverse neural model (12)

Control and Optimization

The basic objective in plant optimization is to control the process variables in order to achieve desired output performance such as better product quality, higher recovery, reduced environment impact, etc. In the past few decades, control techniques have evolved from conventional control (e.g., PID) to model-based control (e.g., Internal Model Control IMC), multivariable predictive control (e.g., Model Predictive Control MPC) and finally adaptive control (e.g., Self-tuning Controller STC and Long Range Predictive Control LRPC) (13).

Instead of working in a real plant, some researchers use dynamic simulators developed with phenomenological modelling approaches to mimic plant processes and to assist in the development of control strategies (14,15,16,17,18). Due to the complex dynamic behaviour of mineral processing plants, a model-based adaptive control device, which will adjust the input variables automatically on-line in such a way as to compensate for process uncertainty, is essential. Neural models can replace conventional dynamic empirical models in model-based control strategies e.g., a neural based predictive control using a forward neural model shown in Figure 5 or a feedforward neural control using an inverse neural model shown in Figure 6 (12). An Inverse Differential Neural Controller (IDNC) that uses first order differentiation of input/output variables to train a network has been reported (19). Examples using neural network controllers with data generated from a dynamic simulator called DYNAFRAG can also be found in the same report.

EXAMPLES

Noise Filtering

A recirculation network (2) is used to remove noise from a set of dissolved oxygen measurements acquired in a flotation enhancement test. The data which consisted of 504 points, were measured and recorded every 3 seconds automatically by a data acquistion system.

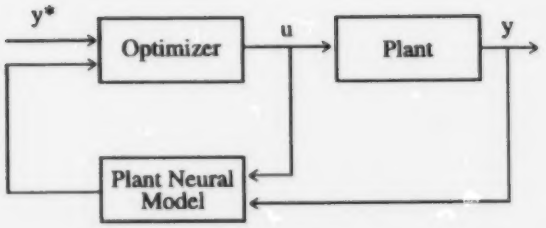


Fig. 5 Neural-based predictive control (12)

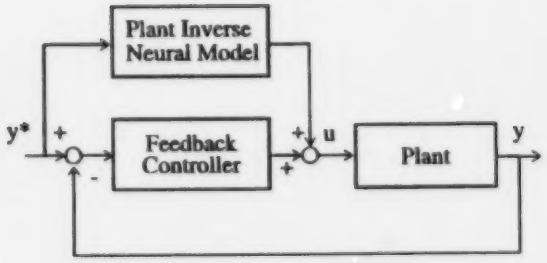


Fig. 6 Feedforward neural control (12)

The network consists of two main layers: the visible layer with 11 Processing Elements (PE) and the hidden layer with 3 PEs as shown in Figure 7. The other two layers: the input and the output layers act as a buffer connecting to the outside world and both have the same number of processing elements as the visible layer. While the visible and hidden layers are fully connected to each other in both directions with adjustable connection weights; the input and output layers are connected through the visible layer in a feedforward direction with fixed connection weights of value 1 only. The network is running in auto-associative mode, i.e., the input signal is to be

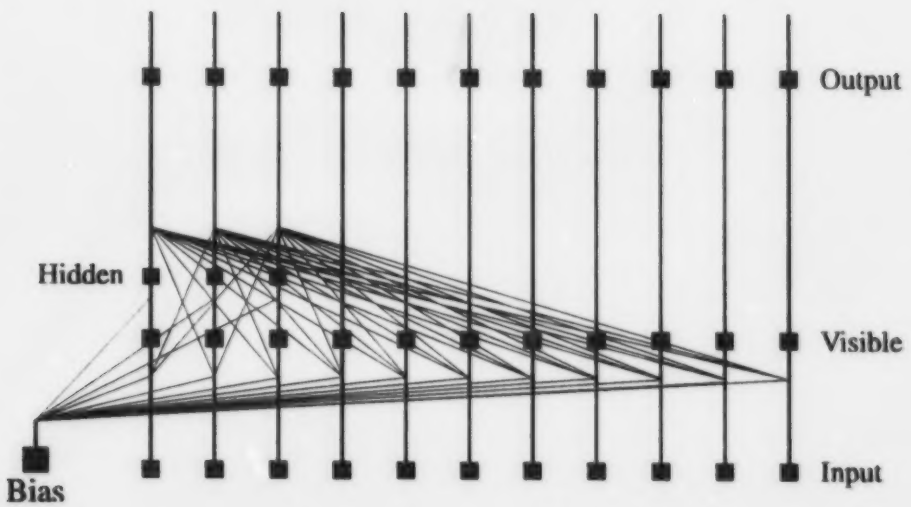


Fig. 7 A recirculation network

reproduced by the network. With a smaller number of hidden Processing Elements (PEs), the hidden layer is trained to perform data compression such that only the first principal components are extracted. The compression ratio, which is defined as the ratio of the number of PEs in the visible layer to the number of PEs in the hidden layer, is used to control how much data connected through the visible layer in a feedforward direction with fixed connection weights of value 1 only. The network is running in auto-associative mode, i.e., the input signal is to be reproduced by the network. With a smaller number of hidden Processing Elements (PEs), the hidden layer is trained to perform data compression such that only the first principal components are extracted. The compression ratio, which is defined as the ratio of the number of PEs in the visible layer to the number of PEs in the hidden layer, is used to control how much data compression one requires in the network design. If the number of hidden PEs is small enough, the network acts as a low pass filter, reconstructing only the most consistent trends in the data.

During training, a moving window of 11 consecutive time samples was repeatedly presented to the network until learning stabilized. The time interval was selected as approximately three times the duration of most of the noise in the signal. The number was chosen such that the network would capture the characteristics of the signals other than the noise and yet remain as simple as possible. After training, only the center processing element of the output layer is used for the output signal and therefore the number of input/output nodes is usually an odd number.

The raw input and filtered output signals as a function of time are shown in Figure 8, while their power spectra are shown in Figure 9. Figure 8 also shows the signal filtered by a moving average method with 5 points on each side of the target point. There are differences between using the moving average method and the NN filtering method especially around time index 375. Although the power spectra in Figure 9 show that the NN filter has eliminated most noise around the zero frequency than the moving average method, it still is not perfect. The NN filtered signal seems to have maintained some roughness of the raw signal.

Several tests were performed by varying the compression ratio, i.e., the number of PEs in the hidden nodes and the size of the time slice presented to the input layer, but no significant

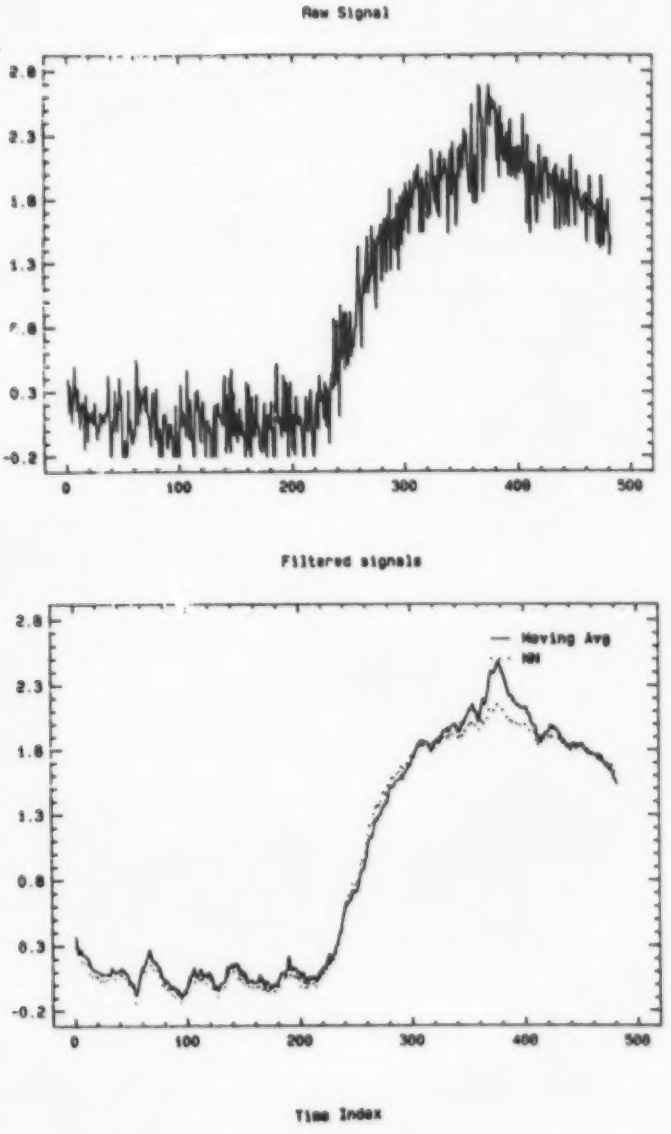


Fig. 8 Raw and filtered signals with NN and moving average

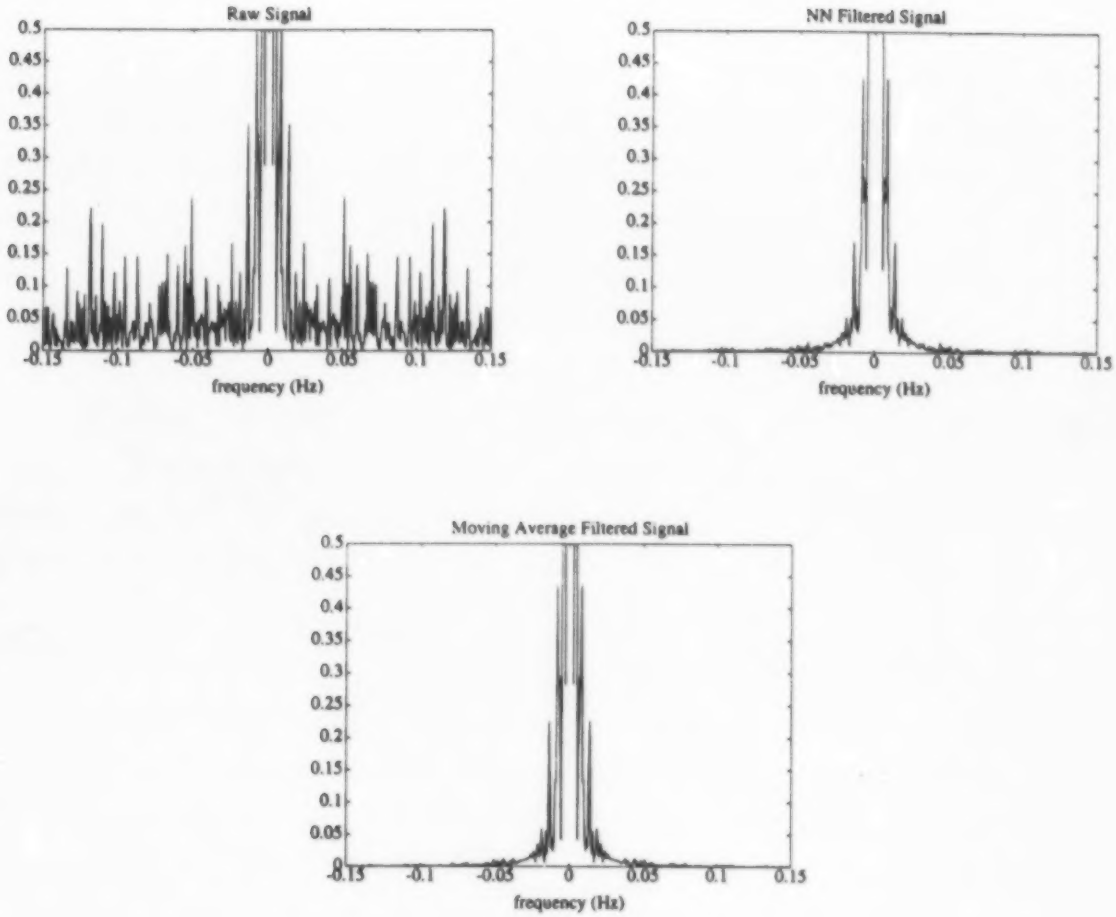


Fig. 9 Power spectra of raw and filtered signals

improvement was observed. More work is required to investigate the potential of using neural networks in low pass filtering applications. It is suggested that more data sets be used to train the network so that there would be at least 50 to 100 data sets for each network connection weight (see Appendix).

Forecasting

This example shows how a neural network (NN) can be used to predict copper concentrates in a copper-zinc concentrator using past values. Data forecasting is useful when current data is

unavailable or erroneous due to mechanical, analytical or process problems. The erroneous data can be identified by an expert system based on known relationship between streams and historical process behaviour and consequently substituted with predicted data (20).

A Back-propagation network (2) which consists of 3 layers is used in this example. There are 12 input processing elements (PE) which respectively represent the present and past copper values at time sequences of t, t-1, t-2 etc. The network is to predict next Cu concentrate at t+1 which is represented by an output processing element. The number of processing elements (PE) in the hidden layer is normally determined empirically and 4 hidden PEs are chosen in this case shown in Figure 10. The network structure is 12-4-1 where the numbers respectively represent the number of PEs in the input, hidden and output layers respectively.

Fig. 10 A back-propagation network used in data forecasting

Figure 11 shows the compared time series of NN calculated concentrates and measured concentrates. Their residuals are also displayed in Figure 11 for comparison. The first 200 data points are used for training the network, whereas the remaining 120 data points are predicted by the model and compared with the given values. The number of elements in the past sequences used in the input layer was varied from 7 to 19 but did not seem to have any effect since most influence comes from the present value at time t.

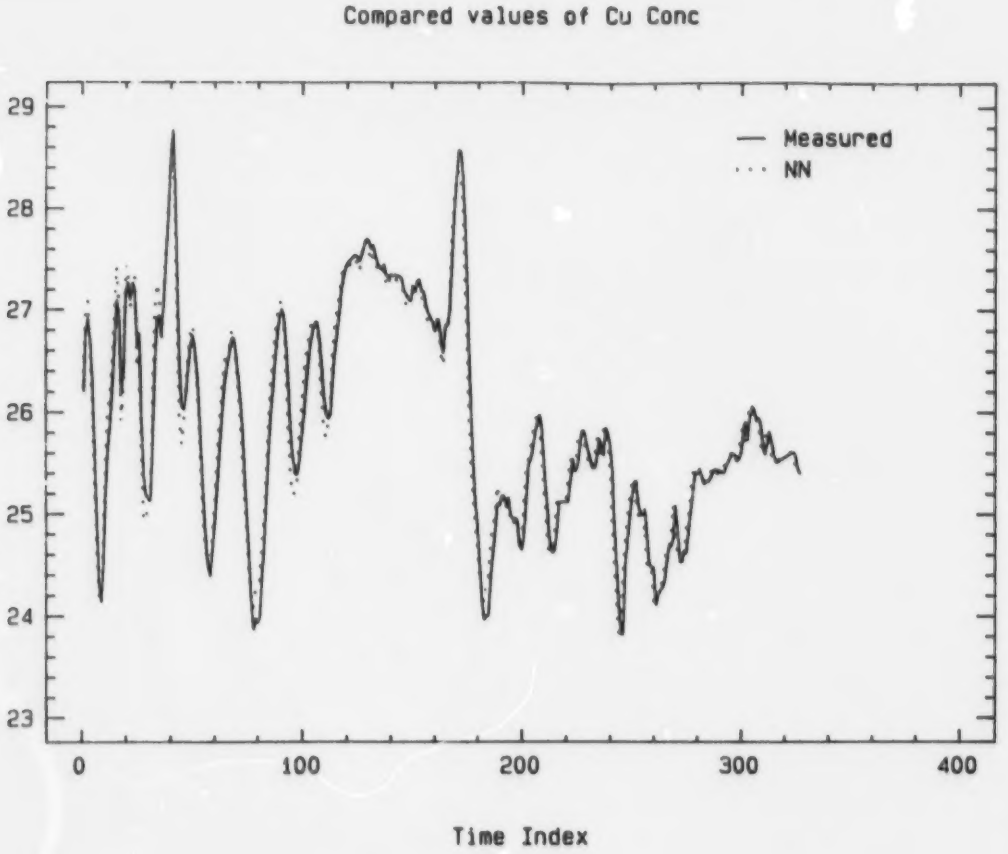


Fig. 11 Data forecasting where the first 200 data points are used for network training and the remaining 120 data points are predicted by network

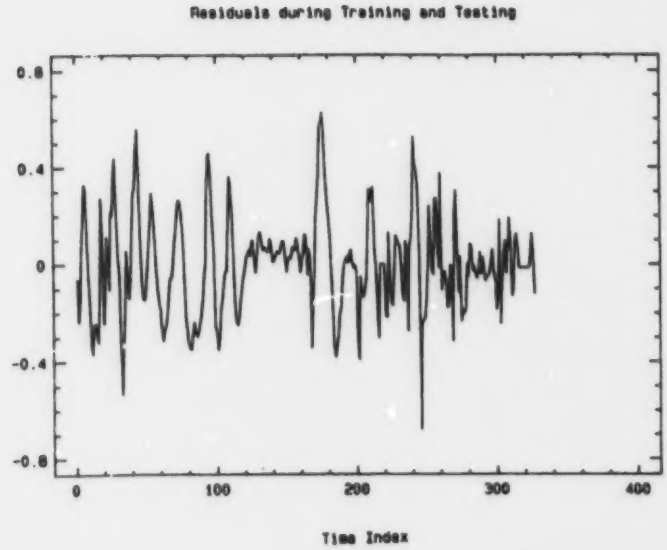


Fig. 12 Modelling residuals of calulated and measured Cu Concentrates

Modelling and Simulation

Process background

Historical data from November 13 to 19, 1989 were provided by Kidd Creek Mines in Timmins, Ontario and the data were collected from their copper-zinc concentrator. Copper and zinc concentrates are produced by flotation separation technique using lime as a pH modifier, 3418A and R208 as copper collectors, R317 as zinc collector, MIBC and DF250 as frothers, and NaCN and SO_2 for sphalerite and pyrite depression (21). Figure 13 shows the process flowsheet for the treatment of the copper-zinc ore which consists of grinding circuits to liberate the minerals, copper flotation circuits to separate the copper minerals and zinc flotation circuits to separate the zinc minerals. 3418A reagent is added in three different locations in primary, secondary rougher and scavenger copper flotation feeds. R208 reagent addition is controlled by a feedforward loop in the rod mill feed in proportional to Head Tonnage with a feedback correction of %Cu in copper concentrates. Copper flotation tails feed the zinc flotation circuit

following reagent addition and conditioning. R317, which is added to the cyclone overflow, zinc rougher and scavenger feeds, is used as a zinc collector.

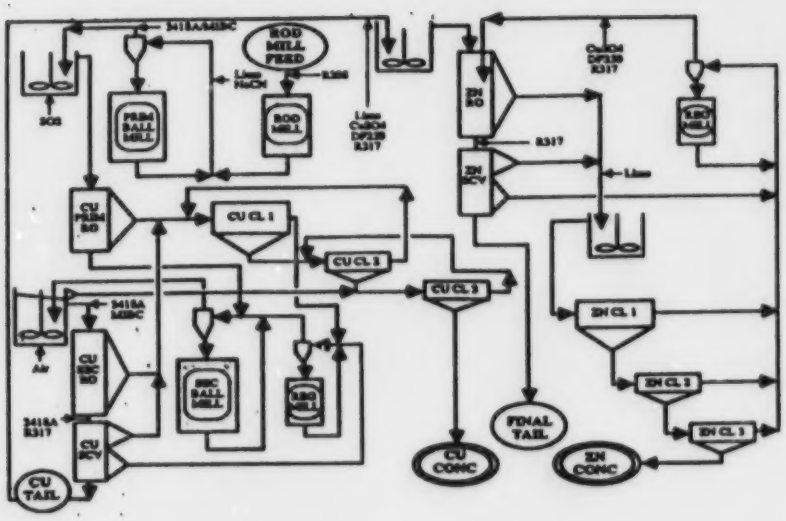


Fig. 13 Process flowsheet for the treatment of copper-zinc ore (21)

Data analysis

A back-propagation network with 3 network layers was used for multivariable identification of the dynamic model of the copper flotation circuit of Division A. A preliminary study using statistical methods was conducted to identify input variables (e.g., reagents used in the circuit, fresh ore feed rate etc.) that have significant effects on output variables (e.g., copper concentrates, grades, recovery etc.). Figure 14 presents the time series of some important variables during the time period of November 14-17, 1989 when the plant was running under normal conditions. The reagent additions were controlled by a computer which was turned on approximately at time index 80. Strategic variable adjustments are applied every 15 minutes following X-Ray Fluorescence analysis, recovery and efficiency calculations. The main control strategy is based on controlling reagent additions to optimize Efficiency/ Grade/ Recovery performance, in addition to a few local control loops installed in rod mill and copper scavenger feeds (21).

In system identification, cross-correlation plots which suggest that some correlation exists between input and output variables are usually used to define the time delays. Figure 15 shows the cross-correlation plots between some variables. There is a maximum correlation between reagent 3418 and Cu recovery at 5 time lags which indicates that Cu recovery relates to the reagent strongly 75 minutes after the addition of the reagent. The other plots do not provide too much information. It appears that head tonnage has very strong but constant correlations with both outputs, positive with %Cu concentrates and negative with Cu recovery. The correlations at negative time lags suggest reverse interactions of those at positive time lags. In other words, the input/output correlation is displayed at positive time lags and output/input correlation is displayed at negative time lags. The almost constant correlations suggest that the variables are independent of each other. There is a strong correlation for R208 and Cu concentrates in the negative time lags which can be explained by the feedback correction used in the copper feed unit control strategy. Auto and cross-correlation functions are usually defined for stationary processes only, however most plant process exhibits nonstationary behaviour, i.e., it does not vary about a fixed mean.

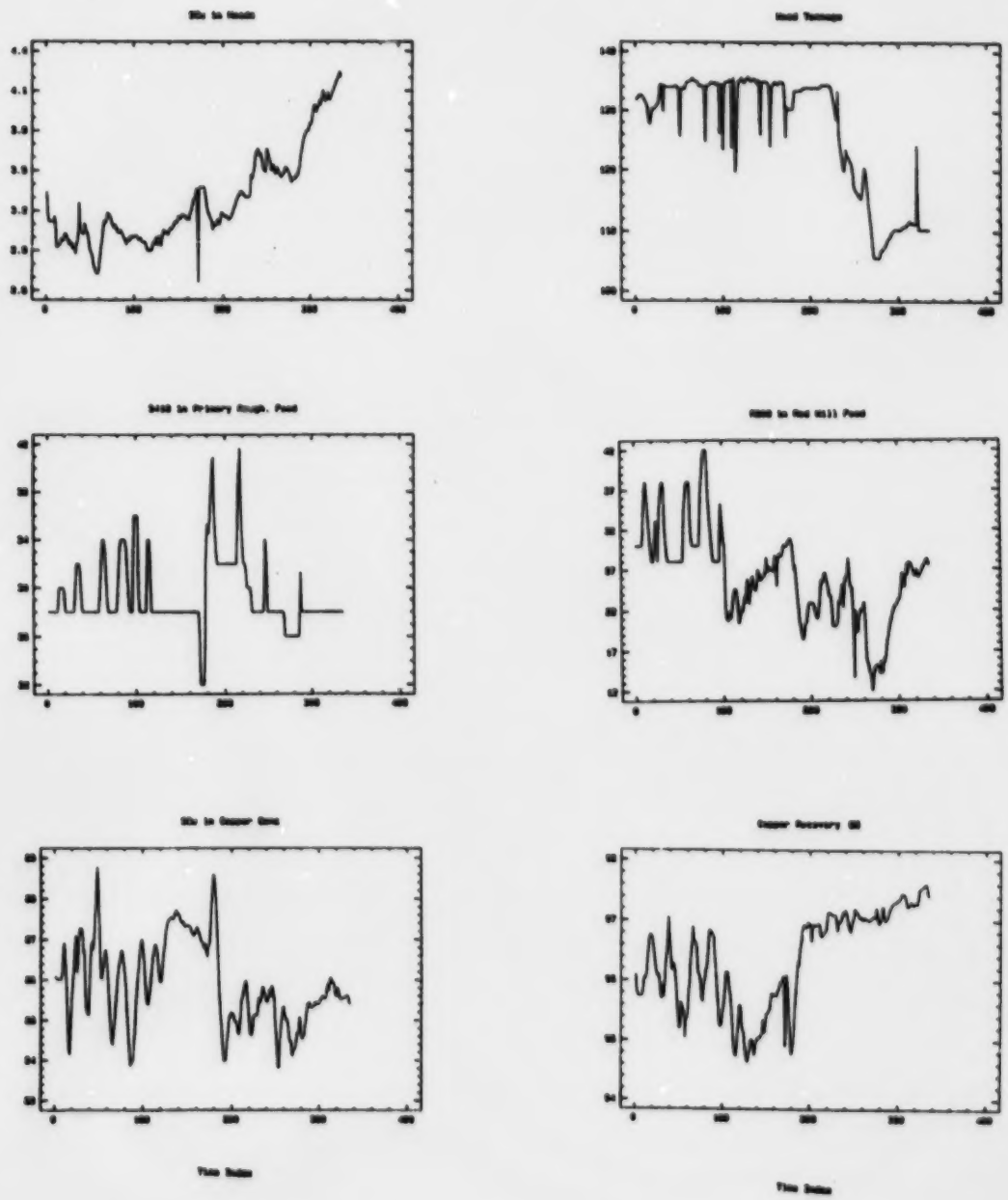


Fig. 14 Time series of input and output variables

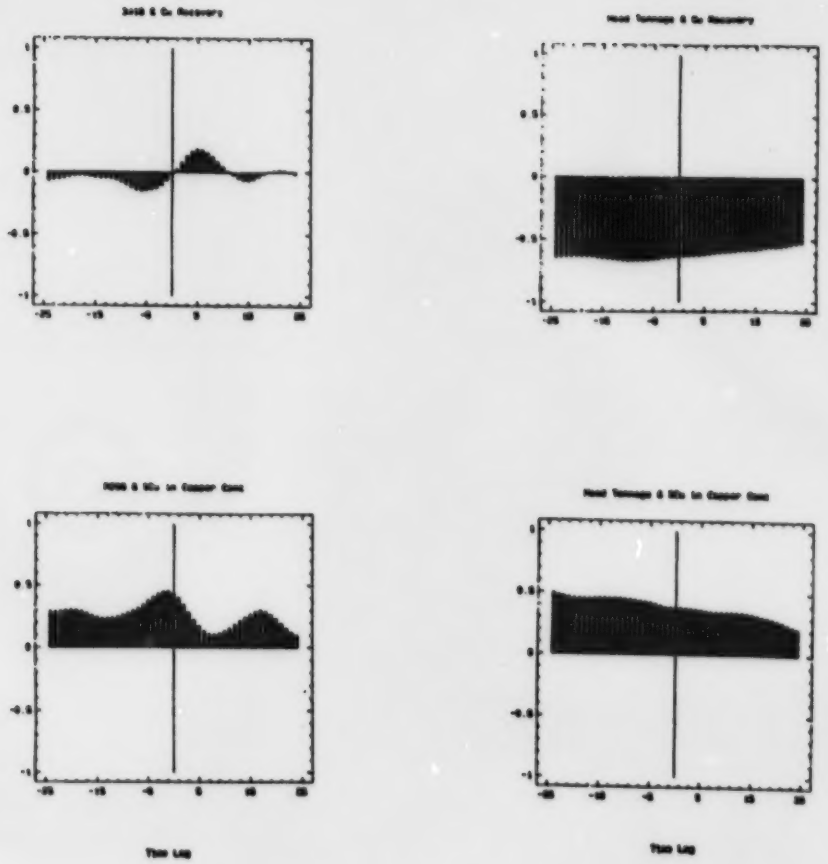


Fig. 15 Cross-correlation plots of some process variables

Stationary behaviour can usually be induced by working with the appropriately differenced time series. The cross-correlation plots of first order differenced time series are shown in Figure 16 which reveal the relationships better. However, no correlations were observed between the head tonnage and the two outputs. The strong correlations between R208 and copper concentrates at

0 time can again be the effect of the feedback correction control strategy (22). Reagent 3418 added in cyclone overflows shows more effect on output variables than those added in rougher and scavenger feeds and thus is the only one chosen in network modelling. Copper efficiency, another performance measurement, is also examined for suitability as an output variable. The analysis shows that there is a similarity with Cu recovery, therefore Cu efficiency is not chosen here.

Partial auto-correlation plots (Figure 17) which gives a measure of the correlation of the same variable at different time lags helps to decide the amount of time lag of a variable one should include as input (23). Table 1 tabulates the results derived from partial auto and cross-correlation functions plotted in Figures 15, 16 and 17. From Table 1a, time delays $\mathbf{d_1}$, $\mathbf{d_2}$ and $\mathbf{d_3}$ of 3,4 and 6 are chosen respectively for 3 input variables %Cu in Heads $\mathbf{U_1}$, reagents 3418 $\mathbf{U_2}$ and R208 $\mathbf{U_3}$ with respect to both output variables copper concentrates $\mathbf{Y_1}$ and copper recovery $\mathbf{Y_2}$.

Network design

An example for this MIMO (Multiple Inputs Multiple Outputs) network model is shown in Figure 18, where 13 input nodes represent the past values of input variables $\mathbf{U_1}$, $\mathbf{U_2}$ and $\mathbf{U_3}$ and output variables $\mathbf{Y_1}$ and $\mathbf{Y_2}$, d1,d2 and d3 are respectively the time delays of $\mathbf{U_1}$, $\mathbf{U_2}$ and $\mathbf{U_3}$, and 2 output nodes represent the 2 present values of outputs $\mathbf{Y_1}$ and $\mathbf{Y_2}$. The time lags of the inputs are chosen in such a way that they include the period where maximum correlations for both output variables have occurred. For example, the 3 time lags for $\mathbf{U_3}$ are **t-6**, **t-7** and **t-8** (with time delay $\mathbf{d_3}$ equals to 6) which also includes the delay time of 7 with output Cu recovery. Not all time delays, e.g., 0 time delay between %Cu in Heads and Cu recovery are used. Head tonnage $\mathbf{U_4}$ is not shown in Figure 18. Throughout the experiments, the network performance improved slightly by increasing the number of PEs from 2 to 9 in the hidden layer. No significant improvement was observed, particularly from 2 hidden PEs to 3 hidden PEs, as was reported in the literature where data was simulated by a dynamic simulator (23). Although a larger number of hidden nodes may produce a better result, the network may also run into the

risk of "memorizing" the training data instead of learning to generalize from the data. Therefore, it is better to use a network with minimal complexity whenever possible. Four PEs was chosen in this example. The network structure is 13-4-2 which represents the number of PEs respectively in the input, hidden and output layers.

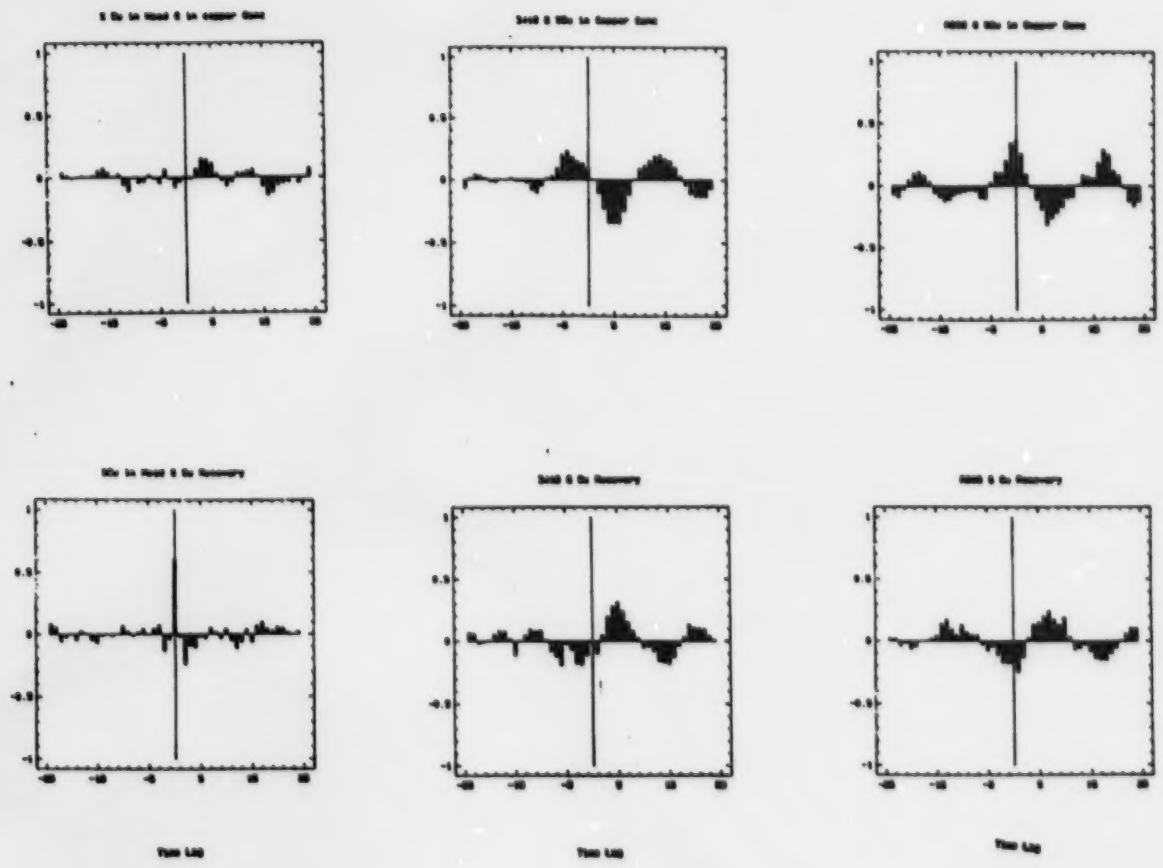


Fig. 16 Cross-correlation plots of main process variables using first order difference of the time series

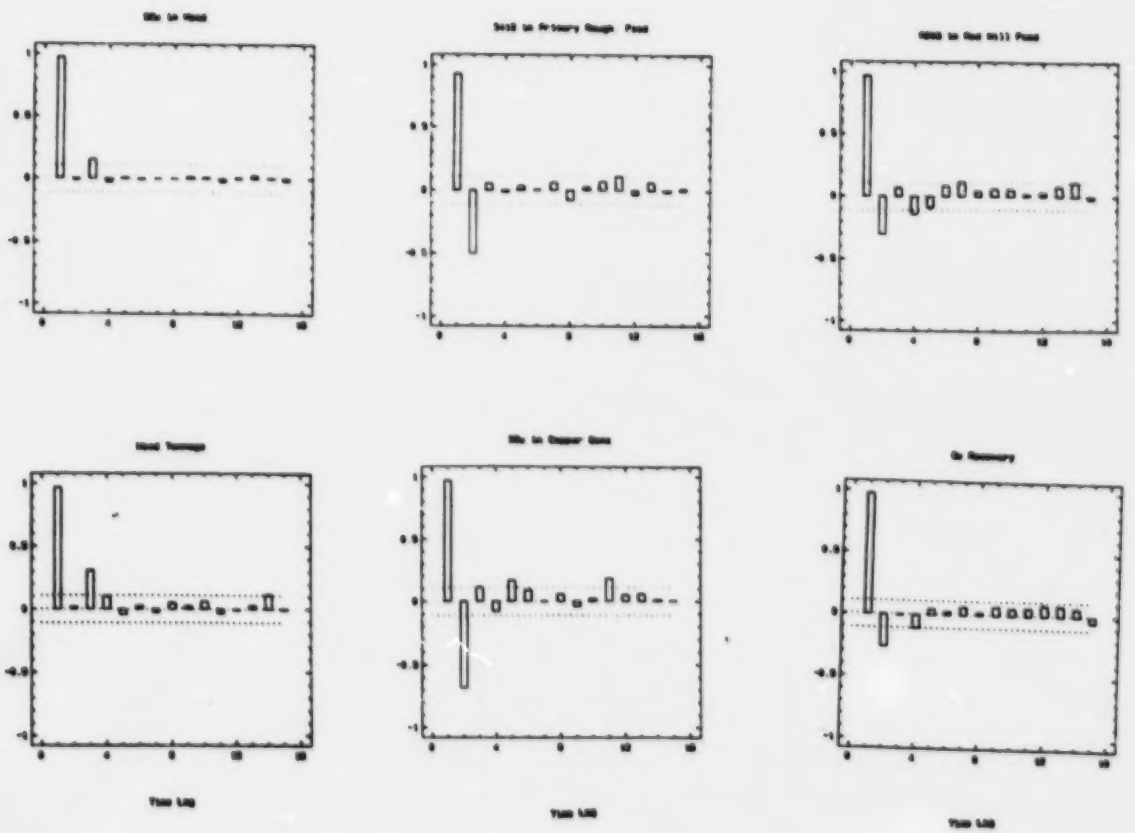


Fig. 17 Partial auto-correlation plots of main process variables

Table 1 a) Time delays determined by cross-correlation functions

Input Variable	Max. Correlation at Time Lag with Output Variable	
	%Cu Concentrates, Y1	Cu Recovery, Y2
%Cu in Heads, U1	3	0, 2
3418APC, U2	4, 5, 6	5
R208, U3	0, 6	1, 7

Table 1 b) Partial auto-correlation of variables

Variables	Max. Auto-correlation at Time Lag
%Cu in Heads, U1	1, 3
3418APC, U2	1, 2
R208, U3	1, 2
Head Tonnage, U4	1, 3
%Cu Concentrates, Y1	1, 2
Cu Recovery, Y2	1, 2

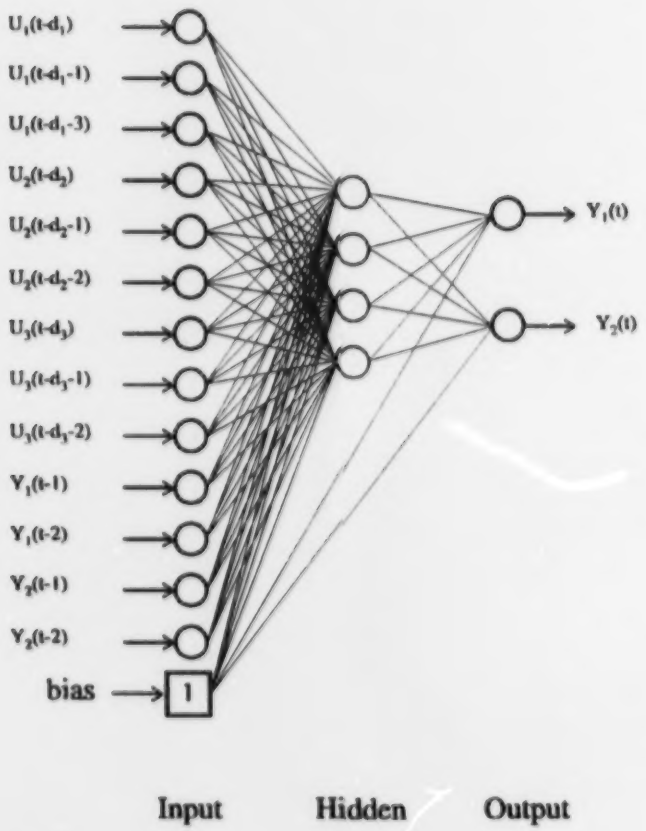


Fig. 18 Input/output node design for MIMO network 13-4-2

Results

a) Network 10-4-2 with 3 input process variables

Using the same data, several tests were performed with a different number of inputs, either by reducing the time lags from each variable or by using different input variables. The idea was to design a network as simple as possible, while maintaining adequate performance. The network chosen here for discussion is similar to Figure 18, except that it has only 10 inputs and the network structure is 10-4-2. The last time lags $t-d_1-3$, $t-d_2-2$ and $t-d_3-2$ from the three input variables: %Cu in Heads, reagents R208 and C3418APC were removed. Figure 19 shows the

comparison of results of the measured and calculated values for the two outputs. The first 200 data points were used for training and the remaining 127 were used for testing or evaluation of the model. Figure 20 shows the scatterplots of the compared results for the two output variables using only the testing data set. It appears that the values simulated by the network model are slightly higher in low copper concentrates and lower in high copper recovery. Figure 21 presents the residuals of the two output variables using both training and testing data sets. Some shifts are observed in the simulation after a certain period - for %Cu, the shift starts at time index 240 i.e., 10 hours after the simulation and for copper recovery, it starts sooner at time index 210 i.e., 2.5 hours after the simulation. There are two reasons that explain why copper recovery deteriorates faster: 1) the test data are outside the range of the data used to train the network; 2) copper recovery is calculated with the copper assays in the head, concentrate and tail streams and the deterioration is compounded by each assay. The large discrepancy in the area between time index 160 and 180 is probably due to the fluctuations occurring in the process where copper assay in heads has dropped suddenly by 20% (see Figure 14). The network model is forced to make adjustment for this sudden change. Actually, the simulated results (for approximately 32 hours, i.e., from time index 201 to 327) are quite satisfactory, the root mean squares (RMS) of the residuals are 0.19 and 0.14 respectively for copper concentrates and copper recovery.

b) Network 12-4-2 with 4 input process variables

Head tonnage was included as one of the input variables in this network model. The network consists of 12 inputs representing previous values of the 4 input variables: Head Tonnage, R208, C3418APC and %Cu in heads and 2 output variables: Cu Concentrates and Cu Recovery; each with 2 time lags. The network structure is 12-4-2. Figure 22 shows the residuals of the two outputs during the training and testing periods. It appears that the residuals of the calculated and measured outputs spreads above and below the zero horizontal more evenly than those in Figure 20. The RMS for the simulated values are 0.21 and 0.15 respectively for copper concentrates and copper recovery. Looking at the Hinton diagram of the trained network, one will notice that the fit of the model is mainly contributed by the nearest previous values of the output variables. A Hinton diagram is a graphical display of the inter-connection matrix with

different colour codes and square sizes to represent different magnitudes of the connection weights and PE values.

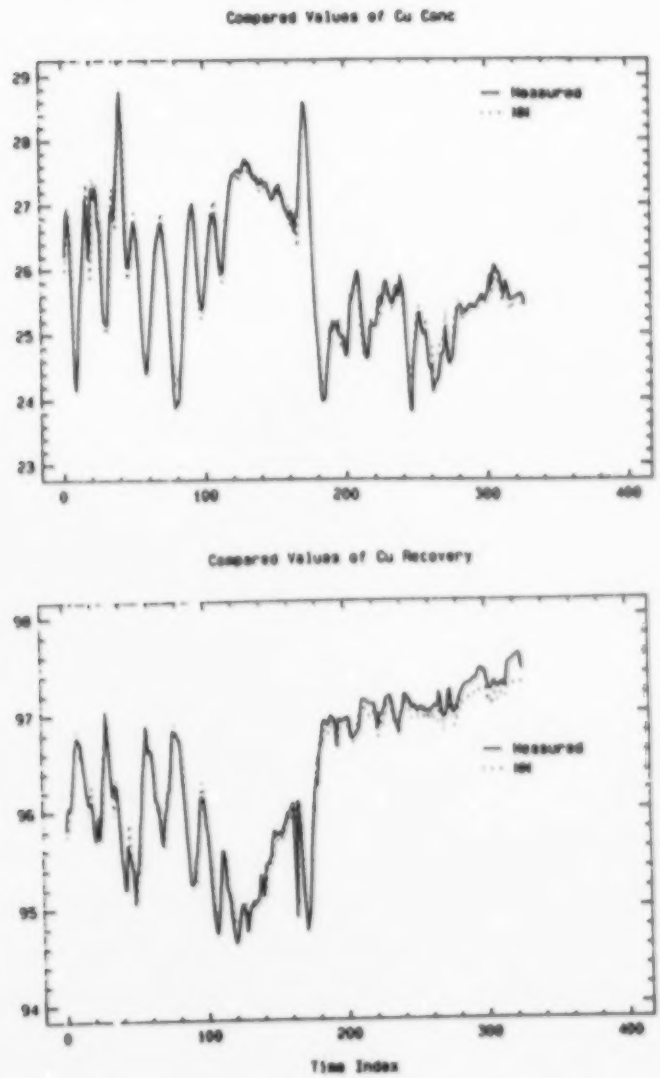


Fig. 19 Calculated and measured values for two output variables where the first 200 data points are used for training the network 10-4-2 and the remaining 127 data points are simulated by the network

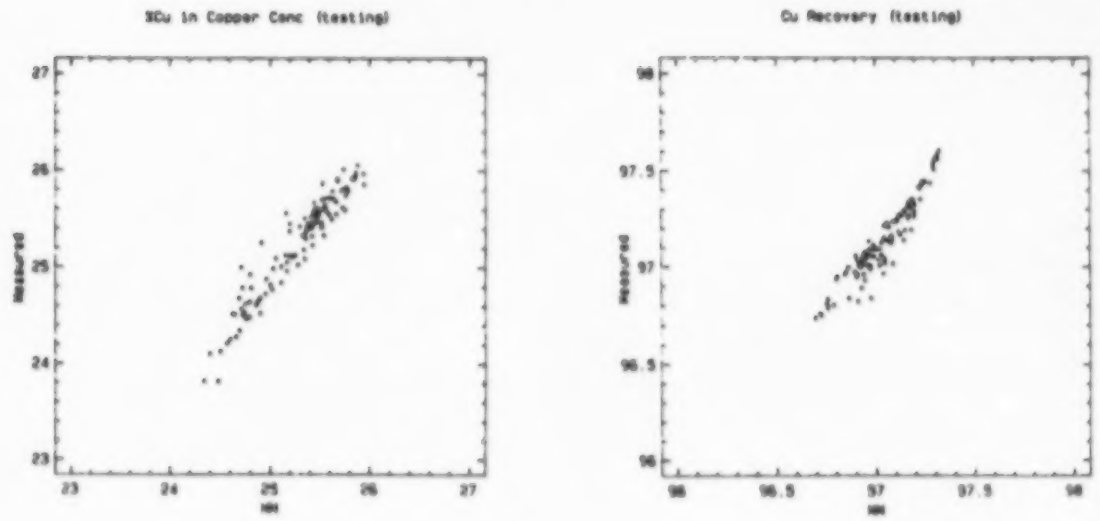


Fig. 20 Scatterplots of two output variables in simulation

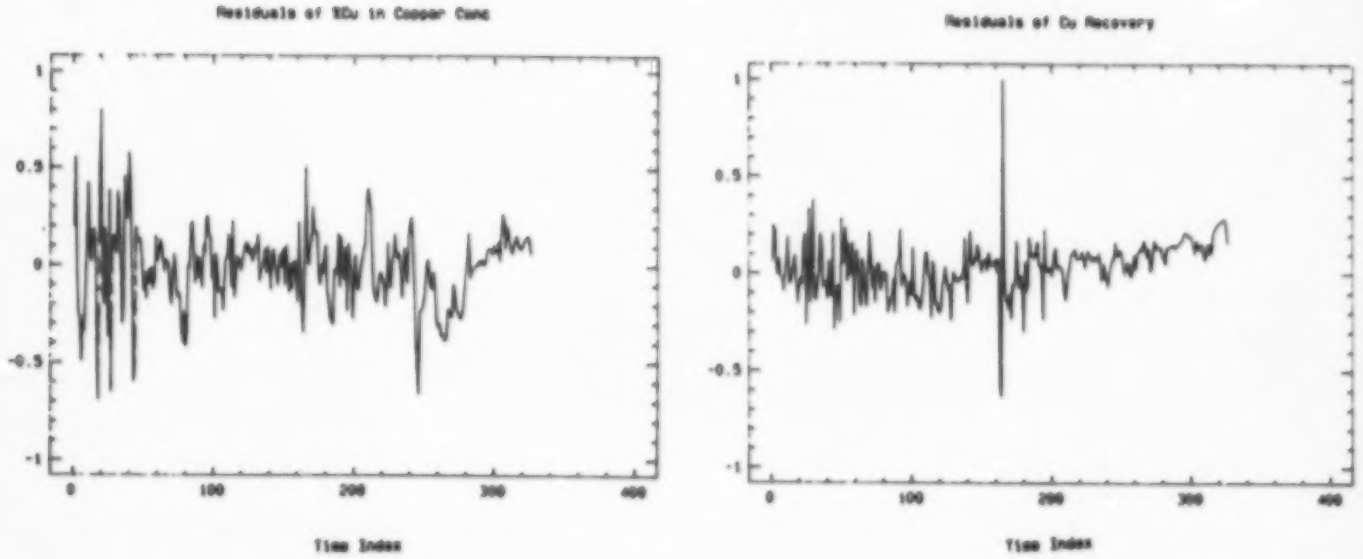


Fig. 21 Modelling residuals of two output variables in modelling and simulation

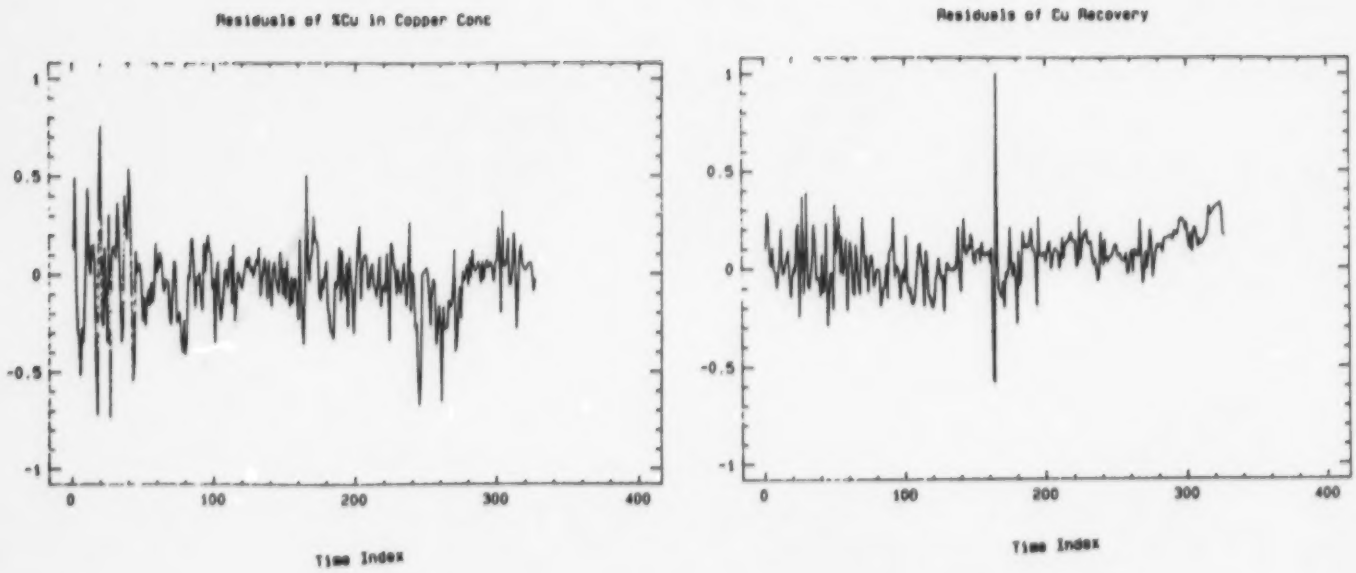


Fig. 22 Modelling residuals of network 12-4-2 where the first 200 data points are used for network training and the remaining 127 data points are simulated by network

Pruning

As was mentioned earlier, one of the advantages of neural computing is its fault tolerance. A network can still function even when a small number of its processing elements are disconnected or disabled. A test was done on the network 10-4-2 above by pruning the network connections with weights less than 10% of the absolute average weights. In other words, the connections with smaller weights are disabled during the training iterations. In this case, most of the

connections were disabled with only a few dominant ones remaining. Their residuals are presented in Figure 23 which shows that the copper assays are worse than copper recovery 5 hours after the simulation. Their RMS values are 1.16 and 0.21 respectively for copper concentrates and copper recovery.

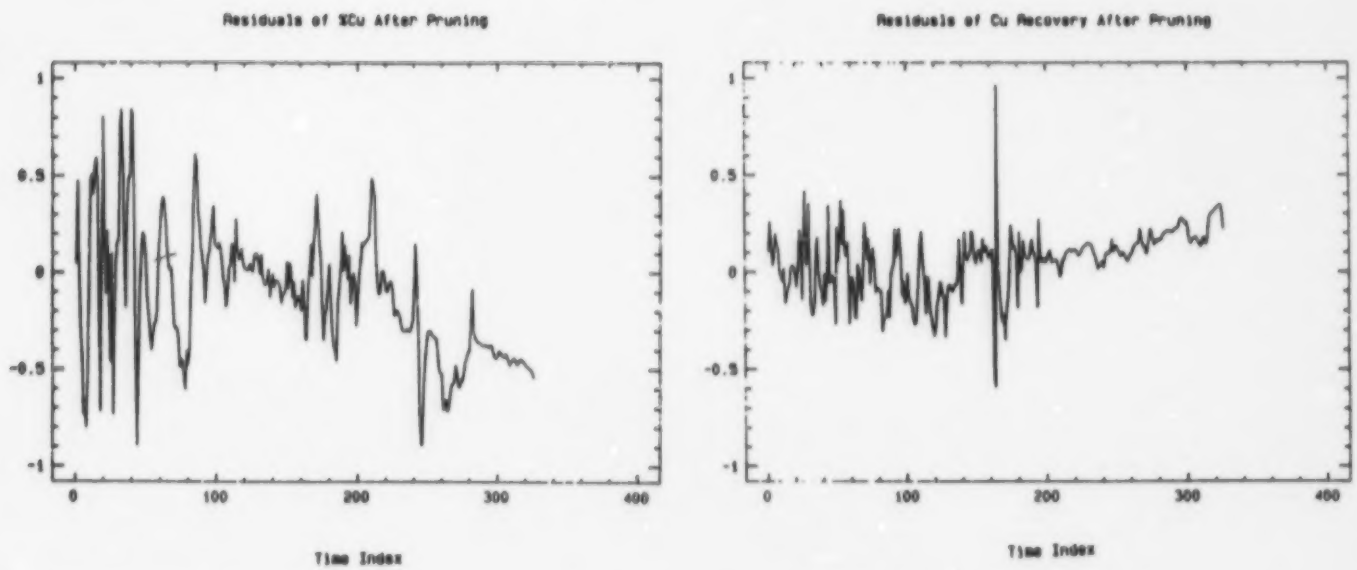


Fig. 23 Modelling residuals of two output variables using network 10-4-2 with 10% pruning

CONCLUSIONS

This report has shown that neural networks can easily be applied to many applications in mineral processing. The examples illustrated here, using laboratory and industrial plant data for noise filtering, data forecasting, modelling and simulation provide confidence that the technology has much to offer. By incorporation with other techniques such as statistical methods for data analysis and performance evaluation, expert systems for making diagnosis and decisions, etc., the application scope for neural networks is further enhanced. The capability to adapt to new information and better fault tolerance makes the technology very attractive to process control applications.

ACKNOWLEDGEMENT

The author gratefully acknowledges E. Ter Heijden, M. Bilodeau and Y. Marcotte for their valuable discussions and reviewing the manuscript and CANMET management for making this project possible.

REFERENCES

1. Lippmann, R.P., " An Introduction to Computing with Neural Nets", IEEE ASSP Mag. April 4-22, 1987.

2. Leung, J., "Neural Networks and their Applications in Mineral Processing Industry", CANMET Division report MSL 92-20 (IR), May 1992.

3. Feigenbaum, E.A. and McCorduk, P., "The Fifth Generation - Artificial Intelligence and Japan's Computer Challenge to the World", Addison-Wesley, 1983.

4. Leung, J. and Laguitton, D., "Latest Developments in Computer Aided Mineral Processing Techniques at CANMET", APCOMIN'90, Application of Computers in the Mining Industry, Santiago, Chile, May 14-17, 1990.

5. Owen, K., "Harry Barrow: Interviewed for Expert Systems", Expert Systems, Vol.7, No. 1, Feb. 1990.

6. Dietz, W., et al., "Jet and Rocket Engine Fault Diagnosis in Real Time", Journal of Neural Network Computing: Technology, Design and Applications, Vol. 1:5-18, 1988.

7. Suddarth, S., et al., "A Symbolic-Neural Method for Solving Control Problems", Proc. of the 1988 IEE International Conference on Neural Networks, San Diego CA, Vol. 1: 515-523, 1988.

8. Saito, K. and Nakano, R., "Medical Diagnostic Expert System Based on PDP Model", Proc. of the 1988 IEE International Conference on Neural Networks, San Diego, CA Vol. 1: 255-262, 1988.

9. Gorman, R.P. and Sejnowski, T.J., "Analysis of Hidden Units in a Layered Network Trained to Classify Sonar Targets", Neural Networks, Vol. 1 pp 75-89, 1988.

10. Lapedes, A. and Farber, R., "Nonlinear Signal Processing Using Neural Networks: Prediction and System Modelling", Los Alamos National Laboratory report LA-UR-87-2662, 1987.

11. "Neural Computing", NeuralWare, Inc., Pittsburgh, PA 15276.

12. Thibault, J. and Grandjean, B.P.A., "Neural Networks in Process Control - A Survey", IFAC Symposium on Advanced Control of Chemical Processes, Toulouse, France, Oct. 14-16, 1991.

13. Fisher, D.G., "Process Control: An Overview and Personal Perspective", The Canadian Journal of Chem. Eng., Vol. 69, Feb. 1991.

14. Hodouin, D., "Dynamic simulators for the Mineral Processing Industry", Computer Applications in the Mineral Industry, Collins & Singhal (eds), Balkema, Rotterdam, 1988.

15. Hodouin, D., et al., "Model Predictive Control of Grinding Circuits", Engineering Foundation Conference on Control of Particulate Processes, Helsinki, Findland, August 1989.

16. Herbst, J.A., et al., "Model-based Control of Mineral Processing Operations", Powder Technology, Vol. 69, No. 1, p. 21-32, Jan. 1992.

17. Stepo, W.D., et al., "Application of Dynamic Process Simulation", presented at South African Council for Automation and Computation Simulation Symposium- Technology's Tool for the Next Decade. Randburg, May 1990.

18. Moloney, M.J., et al., "The Application of Simulation Technology to Enhance Operator Performance", Fourth Mill Operators' Conference, Burnie, Tas 10, March 1991.

19. Thibault, J., et al., "Modelling and Control of Mineral Processing Plants Using Neural Networks", IFAC Workshop on Expert System in Mineral and Metal Processing, Espoo, Finland, August, 1991.

20. Burrows, M.J., et al., "An Expert System Designed to Improve Reliability of Assay Information in a Flotation Plant", Proceedings of the 21st Canadian Mineral Processors Conference, Ottawa, Ontario, January 1989.

21. Thwaites, P., "Computer Controlled Reagent Addition Strategy and Hardware for the Control of a Copper and Zinc Flotation Process", Third meeting of the Canadian Industrial Computer Society, May 1986, Montreal.

22. Bilodeau, M., "Analyse des Series Chronologiques du Circuit de Flottation du Cuivre de Kidd Creek Mines", M.Sc. Thesis, Laval University, June, 1992.

23. Flament, F., Thibault, J. and Hodouin, D., "Potential Applications of Neural Networks for the Control of Mineral Processing Plants", 29th Annual CIM Conference of Metallurgists, Hamilton, August, 1990.

APPENDIX: TIPS FOR NEURAL COMPUTING

A good network model usually results from expertise or accumulated experiences. The factors involved in designing a network model are usually determined empirically, e.g., the number of hidden layers and hidden nodes, epoch value, convergence criteria, number of training iterations, learning coefficients, etc. A summary of useful tips are offered below:

- One hidden layer is adequate for multivariable dynamic modelling. There appears to be no significant improvements in using multiple hidden layers.

- The number of processing elements in the hidden layer should be less than that in the input layer but more than that in the output layer. In noise filtering, the ratio of input nodes to the hidden nodes serves as a compression factor for filtering which is usually considered to be low pass filtering. It is better to use as few hidden nodes as possible to avoid creating an overly complex network. In general, the guidelines (11) are: 1) the more complex the relationship between the input and output variables, the more PEs are required in the hidden layer; 2) there should be at least 5 training data sets for each network connection. For extremely noisy data, this number should jump from 5 to 50 or even to 100, while for very clean data with an exact relationship to the output data, this number could drop to around 2. In the noise filtering example shown above, there are approximately 13 data sets for each active connection weight. This might be a cause factor in the roughness shown in the filtered signal.

- For a large amount of data, the learning time for a Back-propagation network could be quite lengthy. In the example shown in Figure 19, it required at least 20000 iterations in the learning process (i.e., 3 minutes with a Compaq 486 PC) before the combined RMS of the outputs dropped below 0.10 and stabilized. One can use enhanced algorithms such as Extended-Delta-Bar-Delta algorithm (EDBD) (2,11) to improve the

convergence speed. In the tests performed with EDBD, the learning speed was cut down to approximately half, however the fit of the model is slightly off in the high peak region when compared with the normal back-propogation algorithm. The EDBD algorithm can be very useful in a primitive study of new cases for fast system identification and Back-propagation may then be used in fine tuning of the model.

- Convergence performance can be improved by updating the connection weights cumulatively. This is controlled by a value called Epoch which defines the number of data sets accumulated during the learning process before each connection weight update. The epoch value is usually determined empirically: if it is too small, noise in the data set may confuse the network; if it is too large, key trends in the data may be missed. The epoch value used in most of the examples shown is 10.

- Another important factor in training a network is the learning coefficients which are used as scaling factors for the amount of connection weights adjusted between two successive iterations. The use of different learning coefficients for each layer can decrease learning time. In particular, having a larger learning coefficient at the hidden layer than at the output layer allows the hidden layer to form feature detectors during the early stages of training. These feature detectors can then be combined to form more complex detectors at the output layer (11). With large learning rates, a network may go through large oscillations and may never settle or converge (i.e., stop learning). A weight histogram would show that the weights are spreading out to large magnitudes at a rapid rate. This is a sign that the learning coefficients should be reduced. Smaller learning rates, which are required in Back-propagation algorithm, tend to provide more stability in the learning process. Also, as the learning process continues, decreasing the values will cause the network to converge better. However, if the values are too small, it will take too long for the network to converge.

- Sometimes, it may happen that the RMS error of the outputs does not decrease further. The system may be trapped in a local minimum. To get over small bumps, one can jog

the weights both globally (to all connections) and locally (to a specified PE) between a low and high limit. To restart the system from a different starting point, one can initialize and randomize the connection weights. Random initial values for the weights should be smaller for units with many inputs, larger for units with few inputs. This will keep the expected weighted sum within the range of the transfer function (2). Similar to the learning rates, the initial weight values should not be too large, as this might saturate the units and stop the learning process, nor should they be too small, as that might cause the gradients to be very small resulting in slow learning.

- One should not overtrain the network. One way of checking for this condition is to divide the data into two sets: one for training the network and one for testing the network model. The training set for the network should contain data well represented by various outcomes. Usually the longer the training time, the better the model. However, this may only mean that the network can memorize the training data well. The objective in training a network is to make sure that the network will generalize to new input situations. Therefore, the best way to choose the number of iterations required to train a network is to look at the network performance on both training and testing data.